EnergyNN: Energy Estimation for Neural Network Inference Tasks on DPU

Shikha Goel, M. Balakrishnan and Rijurekha Sen
Indian Institute of Technology Delhi, New Delhi, India
E-mail: {shikha.goel, mbala, riju}@cse.iitd.ac.in

***Abstract*—Convolutional Neural Networks (CNNs) are increasingly becoming popular in embedded and energy limited mobile applications. Hardware designers have proposed various accelerators to speed up the execution of CNNs on embedded platforms. Deep Learning Processor Unit (DPU) is one such generic CNN accelerator for Xilinx platforms that can execute any CNN on one or more DPUs configured on an FPGA. In a period of rapid growth in CNN algorithms and the availability of multiple configurations of CNN accelerators (like DPU), the design space is expanding fast. These design points show significant trade-off in execution time, energy consumption and application performance measured in terms of accuracy. To be able to perform this trade-off, we propose a methodology for energy estimation of a CNN running on a DPU. We build an energy model using characteristics of few CNNs and use this model for energy prediction of other unseen CNNs. We evaluate our approach using 16 different standard and popular CNNs with an average prediction error of 9.9%. Energy estimation can be useful in various scheduling applications where one can choose from multiple CNNs based on its energy consumption. We demonstrate the utility of our approach in a drone that is deployed for detecting objects on the ground.**

***Index Terms*—FPGA, CNN accelerator, Deep neural networks, Energy estimation, DPU.**

I. Introduction

High power and resulting energy consumption are some of the biggest concerns in todays' systems, ranging from drones [1] to data-centers [2]. Convolutional Neural Networks (CNN) are used for classification and object detection tasks in such systems. There is a large variety of CNNs which have different power/energy consumption, accuracy and computational/memory requirements as shown in Tables I and IV.

Many prior works are proposed for the acceleration of CNNs or reduction in their power consumption using FPGA [3]–[5]. Deep Learning Processor Unit (DPU) [6] is one such generic accelerator designed specifically for FPGAs by Xilinx, originally developed by DeePhi Tech. and Tsinghua University [4]. It can execute any CNN on any FPGA which can be changed at the run-time by reprogramming the instruction memory i.e. without reconfiguration. DPUs are available in various sizes which differ in the FPGA resources they utilize and their associated computational capacity. Also, multiple CNNs can execute on multiple DPUs concurrently. Such flexibility motivates us to use DPU for our work.

ZCU102 board used in this work was funded by MeitY, Govt. of India under "SMDP-C2SD" project.

Due to the ability of DPUs to execute a variety of CNNs, DPUs allow dynamic switching of CNNs based on the application requirements (at run-time) without the overhead of FPGA reconfiguration. Such dynamic switching is needed for many inference systems like traffic monitoring system [7], MAVI [8] and drones [1]. Due to the availability of large number of CNNs (n) which is further growing continuously, many choices of DPU sizes (m) and possibility of implementing a large number of DPUs on the same device (k), the design space is very large (k^{m^n}). Thus we develop an energy prediction methodology for DPU based systems called EnergyNN (Energy estimation for Neural Network Inference tasks on DPU). It is useful at the design time for rapid design space exploration [9]. It is also useful for an application like drone to choose a set of CNNs during mission planning for suitable deployment at the run-time.

Corcoran et al. [10], Nasser et al. [11] and Lin et al. [12] propose different strategies for estimating the power consumption for FPGAs. Corcoran et al. and Nasser et al. require complete hardware architectural details to estimate the power whereas we propose a strategy where we use very simple features to model energy without any prior knowledge of hardware. Lin et al. [12] propose a hardware based power estimation approach that gets embedded in the hardware itself which is not a very flexible approach. EnergyNN estimates the energy consumption for a particular CNN running on a DPU. It uses basic characteristics of a CNN like its computational and memory requirements to predict the energy consumption. We evaluate our prediction model for 16 different CNNs. In particular, the main contributions of our work are as follows.

1) A prediction model which predicts the energy for any CNN running on a DPU.
2) We show validation of our methodology using actual energy measurements on an FPGA board.
3) We show the use case of our proposed framework for a real world application like drone.

II. Approach adopted for measuring power

In our approach, we first measure power and subsequently use it to measure energy for a CNN running on a DPU. We do layerwise power measurement for a CNN.

A. Why layerwise power measurement?

We measured the power and energy for 16 different CNNs running on a DPU (Table I). The Table shows average PL

TABLE I
POWER, EXECUTION TIME AND ENERGY VARIATION ACROSS VARIOUS CNNS

Network	Execution time (ms)	Power (W)	Energy (mJ)	Number of layers	#MAC operations (x10^8)	Data required (MB)	Type
squeezenet (sq)	1.78	5.18	9.22	26	7.76	4.88	
mobilenet_v2 (mob)	3.75	4.40	16.50	36	6.02	9.86	T
inception_v1 (inc1)	5.23	6.14	32.13	59	31.65	14.62	R
ssd_pedestrian (ped)	8.61	7.54	64.89	35	58.92	17.5	A
resnet50 (res50)	12.03	6.28	75.56	55	77.16	51.98	I
refinedet_1 (ref 1)	24.57	8.04	197.47	48	251.96	41.79	N
vgg16 (vgg)	44.84	5.75	257.89	16	309.41	156.78	
yolo_v3 (yol)	64.01	8.54	546.51	83	605.69	156.95	
resnet18 (res18)	4.42	6.91	30.50	23	36.54	17.08	
inception_v2 (inc2)	6.65	6.09	40.51	79	40.38	22.13	
ssd_adas (adas)	7.75	7.72	59.86	35	62.84	19.8	T
refinedet_3 (ref3)	8.24	7.06	58.23	48	50.85	20.04	E
refinedet_2 (ref2)	12.05	7.59	91.44	48	100.96	25.43	S
ssd_traffic (traf)	12.34	7.96	98.14	35	116.70	21.12	T
ssd_mobilenetv2 (mossd)	15.12	5.88	88.91	50	65.37	43.74	
inception_v3 (inc3)	16.00	6.68	106.87	103	114.26	49.53	

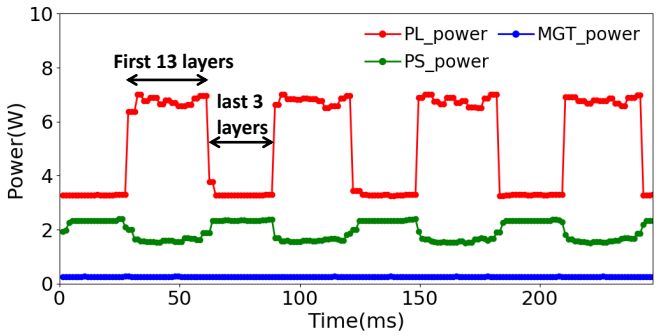

Fig. 1. Variation of power within a CNN (VGG16)

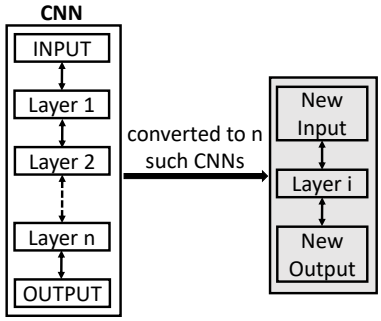

Fig. 2. Breaking a CNN with n layers into n smaller CNNs

(Programmable Logic) power measured for a CNN. The energy values are shown for processing one image. We observe the variations in power values for these CNNs even when running on the same size DPU. To know more about the effect, we plotted the values of power for a particular CNN (Fig. 1). The figure shows the three power components, namely PL (Programmable Logic), PS (Processing System) and MGT (Multi-gigabit transceiver) power. MGT rails are used for transmitting the image to a DisplayPort display so it always remains constant (0.26W) for any CNN.

One of the CNNs, vgg16 with a total of 16 layers, has an execution time of 45 ms. Fig. 1 shows layerwise power consumption for vgg16. Out of 16 layers, first 13 layers are relatively more compute intensive whereas the last 3 layers are more memory intensive. This is reflected in the graph where PL power is more for first 13 layers while for the last 3 layers, PL power drops and PS power slightly increases. Last 3 layers corresponds to fully connected layers where memory bandwidth becomes the bottleneck. Due to this, PS power increases as it includes DDR power. This motivates us to study the layerwise power consumption for different CNNs as power or energy measurements at the CNN level would miss these dynamic patterns within a CNN. The variations in PS power is very small (0.5 W) as compared to variations in PL power as seen in Fig. 1. Hence, for rest of our work, we study only PL power and its corresponding PL energy.

B. Tool and methodology used for power measurement

We use Xilinx FPGA board (ZCU102 [13]) for our measurements and experimentation. Power measurement can be done in a number of ways for ZCU102. The important requirement is we should be able to measure power when CNN is executing on an FPGA.

One of the methods for power measurement is using Maxim's powertool dongle [14]. This dongle can help to measure both current and voltage across FPGA. The limitation for using this dongle is that it does not allow one to record the power measurements at the run-time. Xilinx power estimator (XPE) [15] is another method for power measurement which is a spreadsheet based tool to record the power consumption. This tool can only be used at design time for measurements and not at the run-time. Corcoran et al. [10] reports that XPE has a very high percentage error (624%). Moreover, we predict energy directly instead of separately predicting both power and execution time and then combining. Anyway for battery powered systems like drones energy estimation is a critical requirement.

ZCU102 FPGA board is specifically designed to support power monitoring. It has a total of 18 power rails for the measurement of power across different FPGA components like

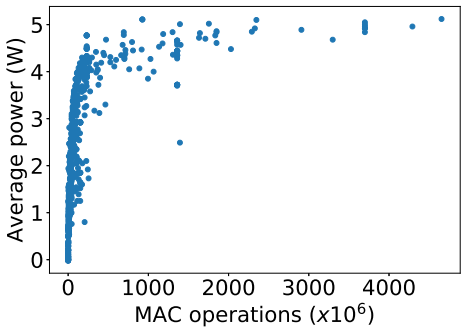

Fig. 3. Relation of MAC operations and power consumption

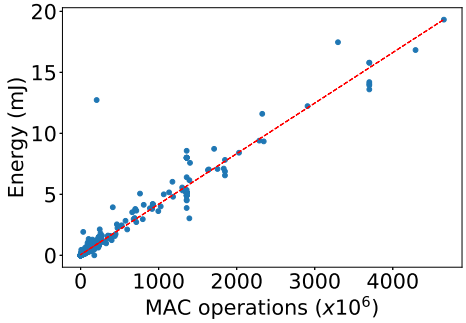

Fig. 4. Relation of MAC operations and Energy

PS and PL, which can be accessed by the application software. This can be useful for power measurements at the run-time when CNNs are executing on an FPGA board. Power from various rails are combined to report PS, PL and MGT power by a Xilinx utility [16]. The minimum sampling rate for power measurement that can be achieved using this utility is 1.3ms.

C. Approach used to measure layerwise power consumption

For our experiments, we considered 16 different CNNs shown in Table I. These CNNs have different execution times which varies from 1.78 ms to 64.01 ms. Within a CNN, most of the layers have execution time less than 1.3 ms. Thus, it is difficult to capture the power for each layer within a CNN if we run the whole CNN together. In our proposed methodology EnergyNN, a CNN is broken into smaller CNNs typically containing just one layer and each smaller CNN is run individually on an FPGA as shown in Fig. 2. For example, if a CNN has 16 layers then it is broken into 16 individual CNNs. The input for the new CNN is modified as the input requirement for that particular layer. We consider a total of 16 CNNs which are broken into 779 new CNNs that run individually to measure the power for each layer in these CNNs. Each new CNN processes 1000 images and is repeated 10 times. We take average of these values to measure total PL power associated with that particular layer. The deviation in measured power values is less than 5% for each layer.

We also observed the effect of input image characteristics on the power consumption of a CNN. We used images that could be considered as extreme cases - A black and white image with fixed pixel values of 0 and 255 respectively, a colour image with considerable variation in pixel values, and an image with 20 objects in it. We observed that the image characteristics do not have any noticeable impact on the power consumption for any CNN. This can be explained as all the CNNs used, run all the computations independent of input characteristics. Power consumption variation due to "data values" seem to be insignificant in DPUs.

III. Energy model, experiments and results

A. Model and feature selection

We started with the prediction of power for a CNN running on a DPU. We plotted the number of MAC operations in a layer vs the power as shown in Fig. 3. We observed that there is no direct correlation between MAC operations and its corresponding power. The graph looks like a scatter plot. On the other hand, we plotted the number of MAC operations in a layer vs energy as shown in Fig 4. We observed that MAC operations and energy in a layer has very high correlation coefficient of 0.98. So, instead of first predicting power and then predicting energy, we predict directly the energy consumed by a layer.

The energy is predicted for each layer in a CNN. These values for individual layers are further summed up to get the energy for the whole CNN. In order to develop an energy model, several features that represent CNN characteristics and DPU architecture were evaluated for prediction. In Table II, we show various features of a layer considered for prediction. We find the correlation coefficient of all the features with energy. We observe that MAC operations and sum of weight, input and output data have maximum correlation factor with energy. Thus, we consider these two features for our prediction model.

The energy of a layer i of a CNN running on a DPU can be written as:

$$Energy(i) = fn\,(MAC_i, Mem_i)$$

- MAC_i represents the number of MAC (Multiply and accumulate) operations a particular layer needs to perform. This feature contributes to the power and energy consumption due to computation happening on the DPU.
- Mem_i represents the memory requirement for each layer. It corresponds to the total amount of data transfer that happens between internal memory (BRAM) and external memory (DRAM). This feature contributes to the power consumption due to data transfers between memories. To calculate the total data transfer, we take sum of number of weight, input and output requirements for each layer.

In each DPU configuration, certain number of MAC operations can be performed in parallel but as each layer typically requires a large number of operations to be performed, a number of such iterations contribute to computation and thus energy of each layer. The ratio of MAC operations to data access is significant as in certain layers input/output requirements may constrain the achievable parallelism in MAC operations.

Our framework use these simple features to predict energy for various type of CNNs (classification or detection task). These features are easily obtained from any CNN descrip-

TABLE II
CORRELATION OF CONSIDERED FEATURES WITH ENERGY

Features considered	Correlation factor with energy
MAC operations	**0.98**
SUM (weight, input and output)	**0.74**
MAX (weight, input and output)	0.64
Outputs	0.53
Inputs	0.49
weights	0.43
Kernel size	0.24
Output image size	0.21
Input image size	0.2
Input channels	0.17
Output channels	-0.09

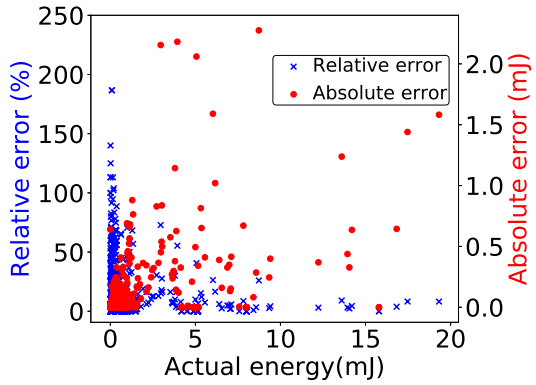

Fig. 5. Percentage error for different layers of various CNNs

tion file. We show results on a specific DPU size (4096 parallel units) which is B4096. This methodology can be easily extended to other DPU sizes by additionally considering DPU specific features for modelling as we considered in our execution time prediction model [17].

B. Experimental setup

To evaluate EnergyNN, we choose 16 different CNNs (total of 779 layers) which have different characteristics like number of computations, number of layers and the data requirement as shown in Table I. Out of these 16 CNNs, we use eight CNNs (*TRAIN* type) to train the prediction model which comprises a total of 358 data points. The rest eight CNNs (*TEST* type), with total of 421 data points, are used to validate the trained model. This choice of CNNs in test and train set is made to distribute a variety of CNNs in both categories.

We use Xilinx Zynq UltraScale+ (ZCU102) FPGA board [13]. We measure PL power for each layer of a CNN on a DPU for 1000 images and take its average value. This average power is multiplied with the execution time for that particular layer to give energy for a layer for processing one image. For a particular CNN and its layers, the number of MAC operations and data requirement information are obtained from the CNN description file.

C. Results

We train our prediction model using various regression models as shown in Table III. The table shows the mean, maximum and median percentage error for these models on

TABLE III
PERCENTAGE ERROR FOR DIFFERENT PREDICTION TECHNIQUES USED

Regression model	Mean Error (%)	Median Error (%)	Maximum Error (%)
Polynomial (degree 2)	9.0	6.1	22.1
Random forest	9.8	10.9	15.6
Linear	**9.9**	**7.0**	**23.6**
Decision tree	14.8	14.3	28.6

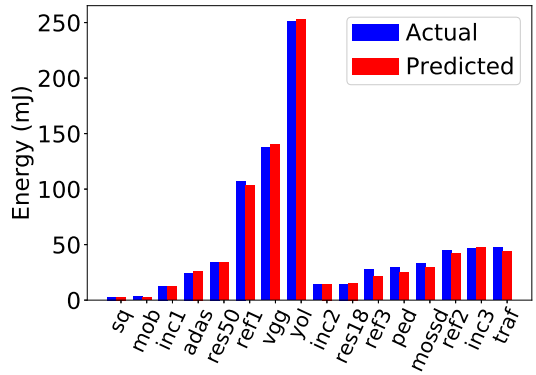

Fig. 6. Actual vs predicted energy for various CNNs running on a DPU

the test dataset. We observe that both linear and random forest have similar mean percentage error. Also, when we go from linear to polynomial regression model, with degree 2, there is a very slight drop in the mean error. Random forest regression model has comparatively higher median error as compared to the other two models. Thus, owing to the simplicity of linear regression model, we tolerate slightly higher errors and choose a linear regression model over other models for our methodology, EnergyNN.

We adopt the sixteen-fold cross validation method. There are a total of 16 CNNs. The linear regression model is trained using 15 CNN models and the model is tested using one CNN. This method is repeated 16 times to consider all possible cases. The average leave-one-out cross-validation error for B4096 DPU is 10.36% whereas the average out-of-sample error for B4096 DPU with 8 train and 8 test CNNs is 9.9%. The high cross validation error is due to some specific CNNs like mobilenet (with depthwise separable convolutions) and resnet (with residual connections) as they have some architectural differences as compared to other CNNs. Thus, we include them in the training data for our prediction model.

Fig. 5 shows the percentage error of energy prediction for different layers of all the CNNs. The figure also shows the corresponding absolute error values for prediction. We observe that most of the layers which have high percentage error in prediction have very small actual energy values (0.01 to 1 mJ). This is because small absolute error results in high percentage error for very small actual values. Absolute error is mostly observed to be less than 2 mJ except for a few outliers. Fig 6 shows the actual and predicted dynamic PL energy to process one image for various CNNs running on a DPU. Dynamic PL energy refers to energy based on dynamic PL power. We observe that energy values varies from 2.86 mJ to 251.19 mJ. The absolute error in prediction varies from 0.00 to 6.54 mJ.

TABLE IV
ENERGY, EXECUTION TIME AND ACCURACY TRADE-OFFS FOR DIFFERENT CNNS

CNN	Execution time (ms)	Accuracy (MAP)	Energy (J)	Flight time of drone (mins)
ssd_mobilenet	14	30	2076	16
refinedet_1	24	67	3316	15
yolo v2	31	78	4713	14
yolo v3 - 320	38	76	5334	14
yolo v3 - 416	61	93	5423	13
yolo v3 - 608	123	94	5500	13

* Table shows PL energy of FPGA when CNN runs for 10 mins.
* Flight time: Time for which drone flies till battery drains out.

IV. APPLICATION OF THE PROPOSED METHODOLOGY

We demonstrate the use case of our methodology for a drone application. We consider the case of a drone which is flying at a fixed speed and a fixed height. The drone has a fixed flight time. The drone has FPGA mounted on it which is used for object detection task. The object detection task is to count the number of vehicles passing by on the road. We use DJI Mini 2 (JP version) Drone [18] specifications for our analysis.

1) Usefulness of EnergyNN in mission planning: Table IV shows the execution time, accuracy and energy tradeoff for various CNNs running on a DPU. Table also shows the flight time of drone when the particular CNN runs on it. The tradeoff is between accuracy and energy. *ssd_mobilenet* would be chosen for energy efficient system as it requires least energy or *yolo v3 - 608* would be chosen if higher accuracy is the preference. Also, as we know that the flight time of drone is fixed - say 15 minutes. Thus the choice would be only between ssd_mobilenet and refinedet_1 as only their energy consumption can support flight time of 15 minutes or higher. (Table IV).

The above example shows how the energy information present in Table IV is useful during the mission planning of a drone to choose between different CNNs. During the time of mission planning, energy values are predicted using EnergyNN and could be stored in a table. Newer CNNs can easily be added without extensive energy measurements using the EnergyNN prediction model.

2) Overhead introduced by EnergyNN: The design space is very large, especially when one can implement a number of DPUs of different sizes. In such scenarios, it is not possible to pre-evaluate energy consumption of all design points and store the values as a table-lookup at the run-time. In such situations, it becomes desirable to predict energy consumption at run-time to enable dynamic decision making. In case of EnergyNN, the time taken to evaluate one CNN choice is of the order of microseconds which is a small fraction of CNN execution time.

3) Design overhead in absence of EnergyNN: Since EnergyNN can predict energy without compiling CNNs or generating bitfiles, it is useful at design-time to choose proper size FPGA and identify suitable CNNs and DPUs as per cost-accuracy trade-off analysis. Doing the same using measurements would take considerable time and effort. For example, if we consider 3 FPGAs, 8 CNNs and 8 DPU sizes, total of 192 bitfiles would need to be generated to evaluate all the choices. Generally, one bitfile generation takes around two to six hours. However, EnergyNN can do such evaluation without generating bitfiles.

V. CONCLUSION AND FUTURE WORK

We presented the motivation for prediction of energy of a Convolutional Neural Network (CNN) running on a hardware CNN accelerator like DPU. We proposed a complete methodology for measurement of energy of a CNN executing on an FPGA. We developed an estimation model which first predicts the dynamic PL energy of each layer of a CNN running on a DPU. These values are summed up to predict energy consumption of the CNN. We evaluated our prediction model on 16 different standard CNNs which gives an average error of 9.9%. We use very simple features like CNN characteristics as the features for energy prediction. Using these simple easy to obtain features and without using any specific hardware implementation details, we can predict energy for any CNN accelerator for DPUs. We show the use case of predicted energy values for a drone application. Once integrated with runtime prediction model, this approach can support many scheduling applications on different platforms. Primarily, one can support the trade-off between execution time, energy consumption and application performance as a future work.

REFERENCES

[1] D. Baek *et al.*, "Battery-aware energy model of drone delivery tasks," in *ISLPED*, 2018.
[2] G. A. Constantinides, "Fpgas in the cloud," in *FPGA*, 2017.
[3] Y. Ma *et al.*, "Alamo: Fpga acceleration of deep learning algorithms with a modularized rtl compiler," *Integration, the VLSI Journal*, 2018.
[4] K. Guo *et al.*, "Angel-eye: A complete design flow for mapping cnn onto embedded FPGA," *IEEE TCAD*, 2018.
[5] Q. Sun, T. Chen, J. Miao, and B. Yu, "Power-driven DNN dataflow optimization on FPGA," in *ICCAD*, 2019.
[6] "DPU for CNN v3.0," 2019. [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/dpu/v3_0/pg338-dpu.pdf
[7] M. S. Chauhan *et al.*, "Embedded CNN based vehicle classification and counting in non-laned road traffic," in *ICTD*, 2019.
[8] R. Kedia *et al.*, "MAVI: Mobility assistant for visually impaired with optional use of local and cloud resources," in *VLSID*, 2019.
[9] R. Kedia *et al.*, "Design space exploration of FPGA based system with multiple DNN accelerators," *IEEE ESL*, in press.
[10] A. Montgomerie-Corcoran, S. I. Venieris, and C. Bouganis, "Power-aware fpga mapping of convolutional neural networks," in *ICFPT*, 2019.
[11] Y. Nasser, J. . Prévotet, M. Hélard, and J. Lorandel, "Dynamic power estimation based on switching activity propagation," in *FPL*, 2017.
[12] Z. Lin *et al.*, "Decision tree based hardware power monitoring for run time dynamic power management in fpga," in *FPL*, 2017.
[13] Xilinx, "Zynq UltraScale+ MPSoC ZCU102 evaluation kit." [Online]. Available: https://www.xilinx.com/products/boards-and-kits/ek-u1-zcu102-g.html
[14] "Maxim integrated." [Online]. Available: https://www.maximintegrated.com/en/products/power/switching-regulators/MAXPOWERTOOL002.html
[15] Xilinx, "Xilinx power estimator," 2020. [Online]. Available: https://www.xilinx.com/products/technology/power/xpe.html
[16] D. Matson and L. Bielich, "Xilinx power measurement," 2019. [Online]. Available: https://developer.xilinx.com/en/articles/accurate-design-power-measurement.html
[17] S. Goel, R. Kedia, M. Balakrishnan, and R. Sen, "Infer: Interference-aware estimation of runtime for concurrent cnn execution on dpus," in *ICFPT*, 2020.
[18] DJI, "DJI Mini 2 drone," 2019. [Online]. Available: https://www.dji.com/mini-2